FPGA based Readout Logic of the Front-end Electronics of the ATLAS Absolute Luminosity Monitor

W. Iwanski[a,b]

[a]CERN, CH-1211 Geneva 23, Switzerland
[b]INP PAN, Cracow, Poland

wieslaw.iwanski@cern.ch

Abstract

Readout of data from front-end electronics of the ATLAS Absolute Luminosity Monitor is controlled by programmable devices.

Alfa-R is a local readout controller which reads digitized data with LHC clock and keeps them until validation of the first level trigger.

Alfa-M is a global readout controller which reads validated events from 23 Alfa-R controllers, forms a data block and sends it to an acquisition system.

In this article, description of logic of both controllers is presented as well as is shown how the controllers can be set up and monitored from an user level.

I. Introduction

The ATLAS Absolute Luminosity Monitor [1][2] consists of 8 Roman Pots, each of them having identical system design [3].

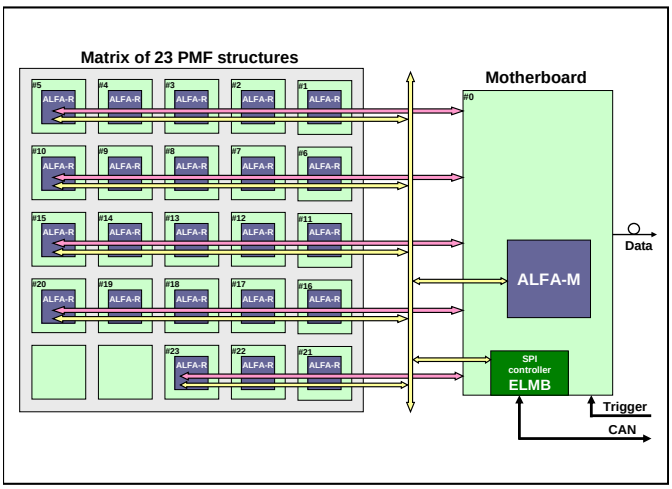

Figure 1: Block diagram of Roman Pot readout sub-system

Readout sub-system of a single Roman Pot consists of a 5x5 matrix of small structures called PMF, each hosting 64 channel Maroc [4] front-end chip and associated with it Alfa-R local readout controller. The PMF is connected via serial links to a motherboard supervised by Alfa-M global controller. This motherboard is interfaced to the triggering system and upon reception of positive decision it transmits front-end data of a given event to further stages of data acquisition system. It provides also a platform for communication with user application to configure and monitor status of electronics. From the user level, readout sub-system is accessible via ELMB[3][5] card which passes user's commands to Alfa-M or Alfa-R controllers onto SPI bus.

Block diagram of one Roman Pot readout sub-system is shown in Figure 1.

II. Local Readout Controller

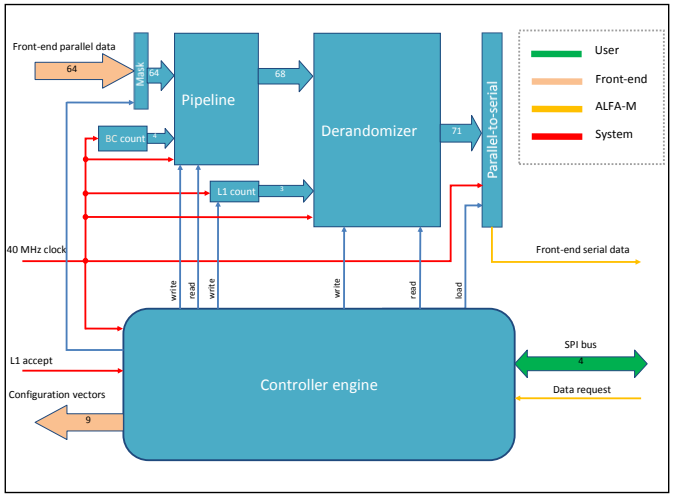

Figure 2: Alfa-R local controller

A. Hardware

Mechanical constraints imposed by space limitations on the PMF structure seriously influenced a choice of the chip hosting Alfa-R logic. In fact, it was only possible to use 256 pins device in fine pitch BGA package. Our choice for prototype version was LC5768MC XPLD chip coming from Lattice.

B. Firmware

Block diagram of the Alfa-R controller logic is shown in Figure 2.

1) Data movement:

On the input, the controller receives 64-bit wide event data from the front-end chip at each 40 MHz LHC clock cycle. Each bit of the input data is AND-ed with respective bit of a mask register to disable faulty input channels. At this stage, masked input data is merged with 3 LSB of the local bunch crossing (BC) counter and written to the pipeline buffer. Function of the local BC counter here is not to give precise information about the time stamp of a certain event but only to make sure that data portions to be merged later by the Alfa-M controller belong to the same event.

The pipeline buffer is 67-bit wide and keeps data as along as it's needed to produce the first level trigger decision (L1) concerning certain event. When decision arrives to Alfa-R, the event data presents on the pipeline output.

Depth of the pipeline is parameterized to allow for accommodation of expected trigger latency. Although in Atlas experiment the L1 decision for each event is produced in fixed time, its latency may vary in different parts of detector electronics as a function of distance. In our design, the trigger latency can be up to 256 clock cycles and it seems to be more than sufficient.

Depending on the L1 decision, the data present on the pipeline output is retained or dropped. Accepted event is merged with 4 LSB of a local L1yes counter and written to the next stage buffer called a derandomizer. Here again, value of the local L1yes counter is used not to fully tag certain event but rather to provide an additional information whether portions of certain event to be merged later are aligned.

The derandomizer is 71-bit wide FIFO buffer used to keep accepted events as long as it is needed to move them from Alfa-R controller to Alfa-M supervisor. It should be known that event is transmitted via serial connection with LHC clock frequency and it requires about 80 clock cycles to move it entirely. Thus, the derandomizer provides storage for all later accepted events while the current one is still being transmitted. In our design depth of the derandomizer is 256 slots which is safe value for 100 kHz rate of expected L1 accepts.

Events stored in the derandomizer are moved always on demand from Alfa-M side. When the latter is ready it sends the Data_Req signal to initiate transmission. Alfa-R in return pushes available event through output shift register to the serial data line together with the Data_Ready signal. The latter tags all valid bits of the shifted event. Data word is shifted out with MSB as first. Only when transmission is finished i.e. the Data_Ready line is dropped by Alfa-R, Alfa-M drops also the Data_Req signal. Violation of this mechanism generates the transmission error made available through the Alfa-R status register.

Timing diagram presenting hand-shaking mechanism during an event transmission is shown in Figure 3.

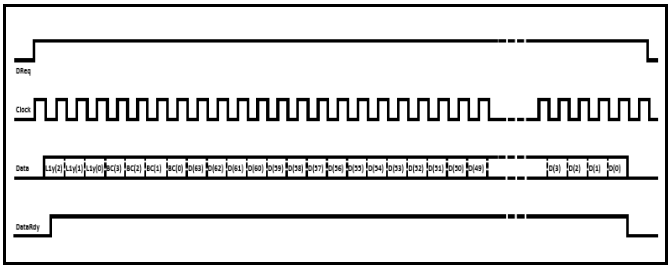

Figure 3: Timing diagram of an event movement from Alfa-R to Alfa-M

2) Control and monitoring:

Control of the front-end as well as monitoring of Alfa-R behaviour is achieved via couple of internal registers implemented in addressing space of Alfa-R. List of registers, their offsets and functions are shown in Table 1.

Table 1: List of internal registers in Alfa-R

Offest	R/W	Register	Description
0x00	W	CONTROL	Control Register
0x01	R	STATUS	Status Register
0x02	RW	MASK1	Mask register 1
0x03	RW	MASK2	Mask register 2
0x04	RW	MASK3	Mask register 3
0x05	RW	MASK4	Mask register 4
0x06	RW	STREAM	Gain and DAC Bitstream Buffer

All registers are 16 bit wide. The write-only control register allows setting internal functions in Alfa-R, while set values are made available via read-only status register. Layout of the control and status registers is shown in Table 2 and 3 respectively.

Table 2: Layout of the control register in Alfa-R

Bit	Control flag	Default	Description
15	TEST_MODE	0	Value of the TEST_MODE flag set in the control register.
14	GAIN_ENABLE	0	Value of the GAIN_ENABLE flag set in the control register.
13	DAC_ENABLE	0	Value of the DAC_ENABLE flag set in the control register.
12	MUX_HOLD	1	Goes to 0 after the 1st write cycle to MUX_CLOCK. Goes to 1 after the 64th write cycle to MUX_CLOCK.
11	RESERVED	-	Reserved flag.
10..5	MUX_CNT	000000	Multiplexer counter.

Table 3: Layout of the status register in Alfa-R

Bit	Control flag	Default	Description
15	TEST_MODE	0	When set to 1, enables transmission of test vectors from Alfa-R
14	GAIN_ENABLE	0	When set to 1, enables the Gain serial port of the Stream register. When set to 0, the Gain serial port outputs are pulled high
13	DAC_ENABLE	0	When set to 1, enables the DAC serial port of the Stream register. When set to 0, the DAC serial port outputs are pulled high
12	MUX_CLOCK	0	Writing a 1 will increase the charge multiplexer counter. The first write cycle will result in the HOLD Status flag pulled low. The Hold flag will remain low until the 64th write cycle occurs

There are 4 mask registers implemented in Alfa-R: (mask1 through mask4) to be able to mask all 64 bits of the input data.

Lastly, the stream register shown in Table 4 is devoted to configure the front-end Maroc chip. As there are no particular requirements for frequency of this configuration, all 6 valid bits of this register are connected directly to configuration

ports of the Maroc device. Writing a certain configuration vector onto those bits, it sets a stable value on concerned ports for a given time slice. This way, writing consecutively a group of vectors one can create desired timing diagram on concerned configuration ports of Maroc.

Table 4: Layout of the stream register in Alfa-R

Bit	Control flag	Default	Description
5	RST_GAIN	0	Value on RST line of GAIN port in Maroc chip.
4	D_GAIN	0	Value on D line of GAIN port in Maroc chip.
3	CLK_GAIN	0	Value on CLK line of GAIN port in Maroc chip
2	RST_DAC	0	Value on RST line of DAC port in Maroc chip.
1	D_DAC	0	Value on D line of DAC port in Maroc chip.
0	CLK_DAC	0	Value on CLK line of DAC port in Maroc chip

For example, writing consecutively patterns “100”, “000”, “110”, “111” on bits 5..3 of the stream register it creates 4 first time slices on lines RST, D and CLK respectively of the timing diagram shown in Figure 4.

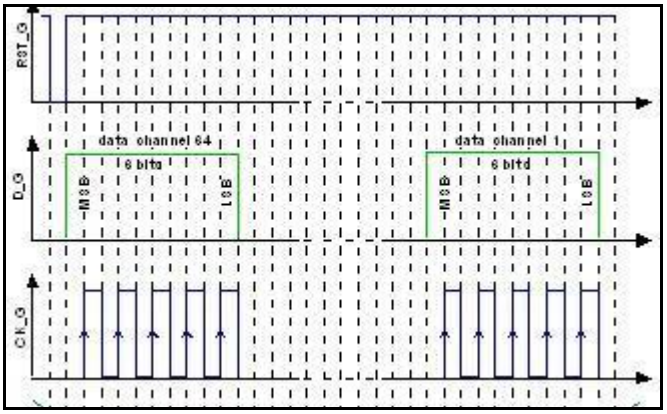

Figure 4: Configuration stream on GAIN port

III. Global Readout Controller

A. Hardware

Alfa-M controller is located on the motherboard [3] where there is much more space available than on the PMF structure. Size of the chip hosting Alfa-M firmware is therefore not critical. Due to high number of I/O signals, 672 pins device in fine pitch BGA package was required. Our choice for prototype version was LC51024MC XPLD chip coming from Lattice.

B. Firmware

Block diagram of the Alfa-M controller logic is shown in Figure 5.

1) Data movement:

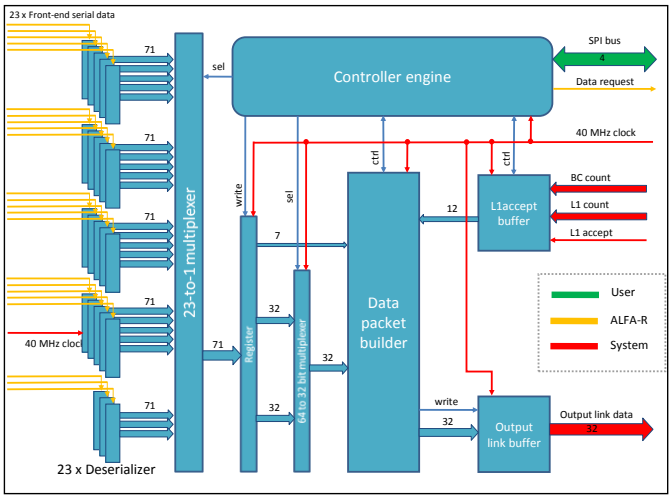

Figure 5: Alfa-M global controller

Controller is interfaced to the triggering system via TTC[3][6] from where it receives L1 accept signal for accepted event. Together with this decision, full length BC and L1 counters states are made available and written to L1 accept buffer.

The L1 accept buffer is a FIFO buffer of 256 slots. Its depth matches the depth of the derandomizer buffers in Alfa-R controllers and its function is to store incoming later L1 accept decisions while the current one is still processed.

L1 accept signal is also delivered to Alfa-R controllers making that the pipeline data is rewritten to the derandomizer buffers (see chapter II, heading B).

Alfa-M triggers readout process for a given event by setting up the Data_Req signal and broadcasting it to all 23 Alfa-R controllers. In return all Alfa-R should in the same time start shifting out 71-bit wide event data words for the requested event (see Figure 3).

All 23 Alfa-R controllers send data via their individual serial link connections to Alfa-M. On the Alfa-M side all serial links are connected to dedicated input shift registers in purpose to de-serialize received data. It will take further about 80 clock cycles before all bits are transmitted.

In the meantime, the data packet builder starts preparation of header for the data block. Prepared words are placed in the output data buffer before input data of the accepted event is fully de-serialized.

All data words from Alfa-R chips are de-serialized after a fixed time interval. Then in each word a logical check is done to see if fractions of BC and L1yes local counters match their equivalents received from the TTC. In case of mismatch a bit is set in a header word of the data block.

Then data words are subsequently multiplexed from first to the last PMF and moved to the output data link buffer. In the end the trailing words of the data block are written to this buffer.

The output data buffer is 32-bit wide FIFO of 256 slots. The data is fetched from there by the serial optical link transmitter [7] as soon as it is stored there and the link is free.

For each event data block of ~2 Kbits is sent. Structure of the data block is shown in Table 5.

Table 5: Layout of the output data block

	WORD	Byte 4	Byte 3	Byte 2	Byte 1
1	SOF	0xB0	0xF0	0x00	0x00
2	BCID	UNUSED[31..12]			BCID[11..0]
3	EVCNT	UNUSED[31..24]	EVCNT[23..0]		
4	PMFERR	UNUSED[31..25]	PMFERR[24..0]		
5	NPMF	UNUSED[31..8]			NPMF[7..0]
6	PMF1L	PMF1[31..0]			
7	PMF1H	PMF1[63..32]			
	…	Next PMFs data			
6 + (NPMF*2)	PARITY	Bitwise exclusive Or of all previous words except SOF			
7 + (NPMF*2)	EOF	0xE0	0xF0	0x00	0x00

2) Control and monitoring:

Similarly to Alfa-R, behaviour of the electronics controlled and monitored by Alfa-M is done via internal registers implemented in addressing space of the controller. Only two 16-bit wide registers were implemented in the prototype version: write-only control and read-only status registers. See the list of registers and their description in Tables 6, 7 and 8.

Table 6: List of internal registers in Alfa-M

Offest	R/W	Register	Description
0x00	W	CONTROL	Control Register
0x01	R	STATUS	Status Register

Table 7: Layout of the control register in Alfa-M

Bit	Control Flag	Default	Description
15	TEST_MODE	0	When set to 1, enables transmission of test vectors from Alfa-M.

Table 8: Layout of the status register in Alfa-M

Bit	Control Flag	Default	Description
15	TEST_MODE	0	1 when the transmission of test vectors from ALFA-R is enabled.
14	GOL_READY	1	1 when the GOL is ready to transmit data.
13	TTC_READY	1	1 when the TTC is receiving a correct signal and is delivering the clock.
12	QPLL_ERROR	0	1 when the QPLL is not initialized.
11	QPLL_LOCK	1	1 when the QPLL delivers a stabilized clock.

IV. Communication via SPI Bus

Communication of user application with any of Alfa-R or Alfa-M controllers occurs via SPI bus. Commands are first sent over CAN bus to the ELMB placed on the motherboard and from there are placed on SPI[3].

SPI data frames are 32-bit wide and are enveloped by the CmdSel signal. Bits 31..16 of the data word carry addressing information and code of operation while bits 15..0 bring the addressed data. Controllers are clients on this bus and are identified by their own local address.

For each bit in the data word, a sampling data strobe is provided on the CmdClk line. Structure of the data word on SPI bus is presented in Figure 6 while meaning of particular fields is presented in Table 9.

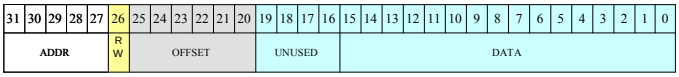

Figure 6: Layout of data word on SPI bus

Table 5: Description of bits in data word on SPI bus

ADDR	R/W	OFFSET	DATA
0: Motherboard **1** to **25**: PMF **31**:Broadcast to all PMFs	**0**: Read **1**: Write	Pointer to a given register in ADDR	Data as specified for addressed register.

When writing from a master to a client i.e. from a user to any local or global controller, all 32 bits of the data word are placed on the CmdIn line.

When reading from controllers to the user, bits 31..16 of the data word are placed on the CmdIn line while the addressed data is expected on the CmdOut line.

Samples of write and read cycles on SPI are shown in Figures 7 and 8 respectively.

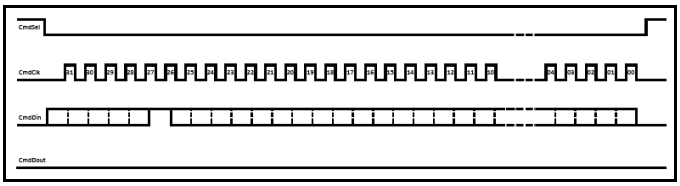

Figure 7: Write cycle on SPI bus

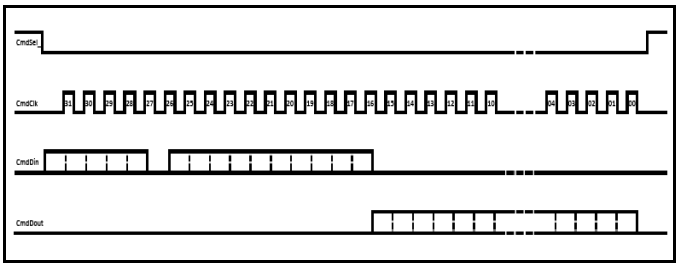

Figure 8: Read cycle on SPI bus

V. Conclusions

Presented in this article readout logic of a single Roman Pot has been successfully tested on the test beam in October 2006[8]. In our prototype set-up, the Alfa-R controller worked with full functionality. Only test vectors generation – feature used for debugging – was available only through a separate firmware. Nevertheless, in order to avoid problems in future upgrades, we will use new type of programmable device for the new PMF design.

Chip working as Alfa-M controller worked with reduced functionality. It proved to be too small to serve all 23 local controllers and its firmware received data only from one row

of them. Also test vectors generation was available through separate firmware. The rest of the logic was working in full scale. Here also new chip will be used in second version of the motherboard.

Programmable devices hosting controllers' logic once again proved to be very practical choice. Designed logic remains fully portable and ready to be used in new bigger devices.

VI. REFERENCES

[1] S. Ask et al., *Luminosity measurement at ATLAS - development, construction and test of scintillating fibre prototype detectors*, *Nucl. Instrum. Meth.* **A568** (2006) 588.

[2] ATLAS Collaboration, *ATLAS forward detectors for measurement of elastic scattering and luminosity determination*, ATLAS TDR, in preparation; *ATLAS forward detectors for luminosity measurement and monitoring*, CERN-LHCC-2004-010, LHCC I-014.

[3] G.Blanchot et al. *System Design of the ATLAS Absolute Luminosity Monitor, Proceedings of the 1st Topical Workshop on Electronics for Particle Physics, Prague, Czech Republic, 3 to 7 Sep 2007*

[4] P. Barrillon et al*., MAROC: Multi-Anode ReadOut Chip for MaPMTs, Proceeding of IEEE – 2006 Nuclear Science Symposium, San Diego, U.S.A., 29 Oct.–2 Nov. 2006.*

[5] B. Hallgren and H. Burckhart*, Front-end I/O via CANBus of the ATLAS detector control system, Proceedings of the 4th Workshop on Electronics for LHC Experiments, Rome, Italy, 21 to 25 Sep 1998.*

[6] P. Gallno*, The ATLAS read out data flow control module and the TTC VME interface production status, Proceedings of the 7th Workshop on Electronics for LHC Experiments, Stockholm, Sweden, 10–14 Sep. 2001.*

[7] P. Moreira et al*., A radiation tolerant gigabit serializer for LHC data transmission, Proceedings of the 7th Workshop on Electronics for LHC Experiments, Stockholm, Sweden, 10 to 14 Sep 2001.*

[8] S. Ask et al., *Hadron beam test of a scintillating fibre tracker system for elastic scattering and luminosity measurement in ATLAS, J. Inst.* **2** No 07 (July 2007) P07004.